EXHIBIT C-2
EXEMPLARY PORTIONS OF PRIOR ART THAT TEACH OR SUGGEST EACH ELEMENT OF THE ASSERTED '661 CLAIMS
PATENT L.R. 3-3(C)

Claim 1 ('661 Patent)	**U.S. 5,249,294 to Griffin et al. ("Griffin")**
A cryptographic processing device for securely performing a cryptographic processing operation including a sequence of instructions in a manner resistant to discovery of a secret by external monitoring, comprising:	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m."

	8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event" Figures 1, 4.
(a) an input interface for receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New

	York, 1997 at 6-7.
(b) a source of unpredictable information;	4:8-16 – "The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) a processor:	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ."
(i) connected to said input interface for receiving and cryptographically processing said quantity,	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ." *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7 (any useful computer must be connected to peripherals such as I/O).
(ii) configured to use said unpredictable information to conceal a correlation between externally monitorable signals and said secret	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution."

during said processing of said quantity by modifying said sequence; and	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:9-10 – "The term 'secure' indicates external inaccessibility." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is competed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number

	of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." Figures 1-2, 4, 5.
(d) an output interface for outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network."

	Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 2 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 1 wherein said input interface and said output interface are the same element.	3:6-10 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term 'secure' indicates external inaccessibility." *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 4 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 1 wherein said cryptographic processing operation includes transforming a message with the Data Encryption Standard (DES).	3:23-27 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine." *See, e.g.*, "Data Encryption Standard," Federal Information Processing Standards Publication (FIPS PUB) 46-2, U.S. Department of Commerce, National Institute of Standards and Technology, Dec. 30, 1993 (suggesting, at 3, implementations of DES; and describing, at 5, high level of protection provided by DES); Menezes, A.J. et al., HANDBOOK OF APPLIED CRYPTOGRAPHY, CRC Press, Boca Raton at 223 and 250 (1997)(describing DES as a well known block cipher and a common element of cryptographic systems).

Claim 6 ('661 Patent)	U.S. 5,249,294 to Griffin
A cryptographic processing device implemented on a single microchip for securely performing a cryptographic processing operation in a manner resistant to discovery of a secret by external monitoring, comprising:	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22– "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION

	subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event" Figures 1 and 4.
(a) an input interface for receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) a source of unpredictable information;	4:8-16 – "The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a

	noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) a processor:	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ."
(i) connected to said input interface for receiving and cryptographically processing said quantity,	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ." *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7 (any useful computer must be connected to peripherals such as I/O).
(ii) configured to use said unpredictable information to conceal a correlation between said microchip's power consumption and said processing of said quantity by expending additional electricity in said microchip during said	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are

processing; and	integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:23-53– "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations." 3:64-4:16– "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. Referring to Figure 2, the loop counter routine includes subroutines 25, 26, 27, 28, 29 and 30. After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data

	processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 7:60-8:24 – "The interim routine of Figure 5 incorporates a common recurring critical routine of the overall larger data processing routine. In this embodiment the critical routine is the movement of source data to a destination. The source data is data that is dynamically processed during the overall larger data processing routine . . . . The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero. The address of the destination of the source data is also stored during the INITIALIZATION subroutine 73." Figures 1, 2, 4, 5.
(d) an output interface for outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7 (any useful computer must be connected to

	peripherals such as I/O). *See generally* W. Rankl and W. Effing, SMART CARD HANDBOOK, John Wiley & Sons, Chichester, 1997 at 356-384 (describing applications for smartcards, including subscriber networks). *See also, e.g.,* Scott Guthery, "Smart Cards," May 28, 1998, www.usenix.org/publications/login/1998-5/guthery.html (visited Dec. 5, 2006) ("Single-chip smart card processors based on these cores are made by almost all the large silicon foundries . . . .Several marketplace forces are at work to open the smart card as a general-purpose computing platform.").

Claim 7 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 6 including program logic to activate said expending during said processing.	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers."\ 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not

	performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is competed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." (emphasis added) 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. Referring to Figure 2, the loop counter routine includes subroutines 25, 26, 27, 28, 29 and 30. After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."

Claim 9 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A cryptographic processing device for securely performing a cryptographic processing operation in a manner resistant to discovery of a secret by external monitoring, comprising:	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution. " 1:37-51 – "The present invention prevents such clock attacks by

	providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a [sic] observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA

	accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event" Figures 1, 4.
(a) an input interface for receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) a source of unpredictable information;	4:8-16 – "The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total

	duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) a processor:	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ."
(i) connected to said input interface for receiving and cryptographically processing said quantity,	3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ." *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7 (any useful computer must be connected to peripherals such as I/O).
(ii) configured to use said unpredictable information to conceal a correlation between externally monitorable signals and said secret during said processing of said quantity;	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers."

	3:9-10 – "The term 'secure' indicates external inaccessibility." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is competed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." (emphasis added) 3:64-4:16– "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. Referring to Figure 2, the loop counter routine includes subroutines 25, 26, 27, 28, 29 and 30. After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the

	selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." Figures 1, 2, 4, 5.
(d) an output interface for outputting said cryptographically processed quantity to a recipient thereof;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(e) a hardware-implemented noise production subunit connected to said source of unpredictable information and configured to expend unpredictable amounts of electricity based on	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred

the output of said source of unpredictable information; and	that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:66-4:2 – "Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. Referring to Figure 2, the loop counter routine includes subroutines 25, 26, 27, 28, 29 and 30." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." Figures 2, 5.
(f) an activation controller, which may be activated by software contained in said device, to activate and deactivate said expending of unpredictable amounts	3:31-39 – "[T]he BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2,..., INTERIM ROUTINE m . . . . To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze." 3:54-63 – "In the sequence of instructions stored in the ROM 18, the next instruction after the instruction for the BRANCH routine 12 is a

of electricity.	TAMPERING DETECTED routine 13. In normal operation, the TAMPERING DETECTED routine 13 will never be executed since the CPU 10 will always branch to the maze of interim routines. However, if an attacker attempts to bypass the BRANCH routine 12, the TAMPERING DETECTED routine 13 will be executed to thereby detect the attacker's tampering attempt." Figures 1, 4.

Claim 10 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 9 wherein said source of unpredictable information is a hardware-implemented random number generator, and wherein said noise production subunit includes a digital-to-analog converter.	4:8-16 – "The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." *See also, e.g.,* English abstracts of JP10084223, JP10197610, JP62260406, and JP62082702 (describing including a digital to analog converter in a noise production subunit);

Claim 11 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A cryptographic processing device for securely performing a cryptographic processing operation in a manner resistant to discovery of a secret by external measurement of said device's power	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution. "

consumption, comprising:	1:36-51 (ADD) – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a [sic] observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of

	the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event" Figures 1, 4.
(a) an input interface for receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) an input interface for receiving a variable amount of power, said power consumption varying measurably during said performance of said	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 2:39-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a [sic] observable external event occurs

operation;	in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 9:18-23 – "The interim routine of Figure 5 need not be executed in combination with other interim routines and may be incorporated into an overall larger data processing routine anytime critical data needs to be moved from one location (source) to another location (destination)."
(c) a processor connected to said input interface for receiving and cryptographically processing said quantity; and	3:9-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7 (any useful computer must be connected to peripherals such as I/O).
(d) a noise production system for introducing noise into said measurement of said power consumption.	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the

	BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable." 3:64-4:6 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 7:52-59 – "FIG. 5 illustrates an alternative preferred, embodiment of an interim routine. In this interim routine the duration between the externally observable event and the predetermined protected routine is randomly varied in response to both dynamically processed data that does not repetitively recur at the same time in relation to each occurrence of the externally observable event, and data stored in a secure memory."

	8:60-9:5 – "The total duration of the interim routine of FIG. 5 is dependent upon the number of ONE bits loaded into the X register during the INITIALIZATION subroutine 73, since the SET iTH BIT OF X TO 1 subroutine 75 is executed only when the ith bit of the X register equals ONE. Thus the range of variation in the duration of the interim routine of FIG. 5 is the number of clock cycles required for the subroutine 75 times the number of bit positions in the X register. Since the number of times that the subroutine 75 is executed is dependent upon the content of the dynamically processed source data and the secure data in the RAM, the total duration of the interim routine of FIG. 5 is a random variable that is unknown to the observer." Figures 2, 5.

Claim 12 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 11 wherein said noise production system comprises: (a) a source of randomness for generating initial noise having a random characteristic;	8:60-9:5 – "The total duration of the interim routine of FIG. 5 is dependent upon the number of ONE bits loaded into the X register during the INITIALIZATION subroutine 73, since the SET iTH BIT OF X TO 1 subroutine 75 is executed only when the ith bit of the X register equals ONE. Thus the range of variation in the duration of the interim routine of FIG. 5 is the number of clock cycles required for the subroutine 75 times the number of bit positions in the X register. Since the number of times that the subroutine 75 is executed is dependent upon the content of the dynamically processed source data and the secure data in the RAM, the total duration of the interim routine of FIG. 5 is a random variable that is unknown to the observer."
(b) a noise processing module for improving the random characteristic of said initial noise; and	9:6-16 – "The randomness of the the total duration of the interim routine of FIG. 5 is further compounded by execution of the WRITE NEW SECURE DATA subroutine 79, which changes the secure data in the RAM that is accessed during the INITIALIZATION subroutine 73, so that each time the interim routine of FIG. 5 is executed, such secure data may be different. The secure data may be provided in the RAM in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) a noise production module configured to vary said power	9:6-16 – "The randomness of the the total duration of the interim routine of FIG. 5 is further compounded by execution of the WRITE NEW SECURE DATA subroutine 79, which changes the secure data

consumption based on an output of said noise processing module.	in the RAM that is accessed during the INITIALIZATION subroutine 73, so that each time the interim routine of FIG. 5 is executed, such secure data may be different. The secure data may be provided in the RAM in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."

Claim 13 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The device of claim 12 wherein said noise production system is connected to said processor and is selectively operable under the control of said processor.	9:6-16 – "The randomness of the the total duration of the interim routine of FIG. 5 is further compounded by execution of the WRITE NEW SECURE DATA subroutine 79, which changes the secure data in the RAM that is accessed during the INITIALIZATION subroutine 73, so that each time the interim routine of FIG. 5 is executed, such secure data may be different. The secure data may be provided in the RAM in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."

Claim 22 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A device according to claims 1, 4, 7, 9, 11, 14, 15, or 20 wherein said device comprises a smartcard.	1:11-18 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'. A clock attack is a procedure by which an attacker gains access to secure data or code used in a predetermined data processing routine being executed within a secure data processor, such as a secure microprocessor . . . ." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." *See generally* W. Rankl and W. Effing, SMART CARD HANDBOOK, John Wiley & Sons, Chichester, 1997 at 356-384 (describing applications for smartcards, including subscriber networks).

Claim 23 ('661 Patent)	U.S. 5,249,294 to Griffin
A method of securely performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring, comprising:	1:11-21 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'. A clock attack is a procedure by which an attacker gains access to secure data or code used in a predetermined data processing routine being executed within a secure data processor, such as a secure microprocessor, by determining the time of execution of the predetermined data processing routine in relation to occurrence of an observable external event that precedes the predetermined routine . . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution. " 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external

	inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event"
(a) receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of

	pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) generating unpredictable information;	3:12-22 – "CPU 10 . . . further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) cryptographically processing said quantity, including using said	1:29-34– "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of

unpredictable information while processing said quantity to conceal a correlation between externally monitorable signals and said secret by selecting between:	pins which is related to or affected by internal processor execution." 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:9-10 – "The term 'secure' indicates external inaccessibility." 3:23-35 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable." 3:64-4:66 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the

	number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." Figures 1, 2, 4, 5.
(c)(1) performing a computation and incorporating the result of said computation in said cryptographic processing, and	3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding

	instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is competed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1."
(c)(2) performing a computation whose output is not incorporated in said cryptographic processing; and	3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is competed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly)

	random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable."
(d) outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 24 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The method of claim 23 where said selecting is performed in software.	3:11-22 – "Referring to FIG. 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines ROUTINE N-1, BRANCH, ROUTINE N, TAMPER DETECT AND ROUTINE N+1 (respectively identified by reference numerals 11, 12, 14, 15, 16) in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, . . . , INTERIM ROUTINE , (respectively identified by reference numerals 20, 21, 22) assembled in response to instructions accessed from a secure RAM 24." 6:27-48 – "Referring to FIG. 4, in an alternative preferred embodiment of the present invention that requires less secure RAM

	capacity than the preferred embodiment of FIG. 1, a CPU 44 executes a group of data processing routines ROUTINE N-1, BRANCH, ASSEMBLE AND EXECUTE INTERIM ROUTINES, ROUTINE N AND ROUTINE N+1 (respectively identified by reference numerals 46, 47, 48, 49, 50) in response to instructions (code) accessed from a ROM 51. The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an externally observable event during each sequence of the overall larger routine. In order to prevent the routine ROUTINE N from being synchronized with the externally event that repetitively precedes the routine ROUTINE N, the BRANCH routine causes the CPU 44 to branch to a maze of m interim routines assembled from a group of n interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, . . . , INTERIM ROUTINE n (respectively identified by reference numerals 54, 55, 56) stored in the ROM 51."

Claim 25 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The method of claim 23 where said selecting is performed in hardware on an integrated circuit including a microprocessor.	3:4-7 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit." 3:11-22 – "Referring to FIG. 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines ROUTINE N-1, BRANCH, ROUTINE N, TAMPER DETECT AND ROUTINE N+1 (respectively identified by reference numerals 11, 12, 14, 15, 16) in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, . . . , INTERIM ROUTINE , (respectively identified by reference numerals 20, 21, 22) assembled in response to instructions accessed from a secure RAM 24." 6:27-48 – "Referring to FIG. 4, in an alternative preferred embodiment of the present invention that requires less secure RAM capacity than the preferred embodiment of FIG. 1, a CPU 44 executes a group of data processing routines ROUTINE N-1, BRANCH, ASSEMBLE AND EXECUTE INTERIM ROUTINES, ROUTINE N AND ROUTINE N+1 (respectively identified by reference numerals 46, 47, 48, 49, 50) in response to instructions (code) accessed from a ROM 51. The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the

	occurrence of an externally observable event during each sequence of the overall larger routine. In order to prevent the routine ROUTINE N from being synchronized with the externally event that repetitively precedes the routine ROUTINE N, the BRANCH routine causes the CPU 44 to branch to a maze of m interim routines assembled from a group of n interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, . . . , INTERIM ROUTINE n (respectively identified by reference numerals 54, 55, 56) stored in the ROM 51." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network."

Claim 26 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A method of securely performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring, comprising:	1:11-21 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'. A clock attack is a procedure by which an attacker gains access to secure data or code used in a predetermined data processing routine being executed within a secure data processor, such as a secure microprocessor, by determining the time of execution of the predetermined data processing routine in relation to occurrence of an observable external event that precedes the predetermined routine . . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution. " 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles."

	2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim

	routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event"
(a) receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) generating unpredictable information;	3:12-22 – "CPU 10 . . . further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable."

	7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) cryptographically processing said quantity, including using said unpredictable information while processing said quantity to conceal a correlation between externally monitorable signals and said secret	1:29-34– "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:9-10 – "The term 'secure' indicates external inaccessibility." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In

	order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable." 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." Figures 1, 2, 4, 5.
by selecting a code process from a plurality of code processes, where said selected code process	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention,

is involved in said cryptographic processing,	steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an externally observable event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the externally observable event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, . . . , INTERIM ROUTINE m. The total duration of the m interim routines is a random variable. The m interim routines have different durations. To ensure that the branch into the maze of interim routines is taken, some subroutine critical to the overall larger routine is buried inside the maze. If this subroutine is not performed, then the executed overall larger routine is not valuable. The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is completed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are

	accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
but where the value of said outputted quantity is independent of which of said code processes was selected; and	3:41-53 – "The critical subroutine is buried by breaking up the instructions in such a way that the instruction for the predetermined protected routine ROUTINE N does not reside sequentially after its preceding instruction. Instead, the BRANCH routine 12 saves the address of the instruction for the predetermined protected routine ROUTINE N in a secure memory location that cannot be accessed until after execution of the maze of interim routines 20, 21, 22 is completed. After the address of the instruction for the predetermined protected routine ROUTINE N is stored, the CPU 10 executes the first interim routine INTERIM ROUTINE 1." 6:1-6 – "Upon completing execution of the maze of interim routines, the CPU 10 executes the protected routine ROUTINE N after accessing the address of the instruction of the protected routine ROUTINE N, which was saved by the CPU 10 when the CPU executed the BRANCH routine 12." 7:65-68 – "The interim routine of Figure 5 is executed between each occurrence of the observable external event and the execution of the predetermined protected routine." 9:18-22 – "The interim routine of Figure 5 need not be executed in combination with other interim routines and may be incorporated into an overall larger data processing routine anytime critical data needs to be moved from one location (source) to another location (destination)." Figure 5.
(d) outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber

	communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 27 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A method of securely performing a cryptographic processing operation including a sequence of instructions in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring, comprising:	1:11-22 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'. A clock attack is a procedure by which an attacker gains access to secure data or code used in a predetermined data processing routine being executed within a secure data processor, such as a secure microprocessor, by determining the time of execution of the predetermined data processing routine in relation to occurrence of an observable external event that precedes the predetermined routine . . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed

	from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event"
(a) receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a

	descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(b) generating unpredictable information;	3:12-22 – "CPU 10 . . . further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM 24." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(c) using said unpredictable information while processing said quantity to conceal a correlation between externally monitorable signals and said secret by using said	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution. " 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to

unpredictable information to modify said sequence; and	occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:9-10 – "The term 'secure' indicates external inaccessibility." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable." 3:64-4:16 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27. The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are

	accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."
(d) outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 29 ('661 Patent)	**U.S. 5,249,294 to Griffin**
A method of securely	1:11-22 – "The present invention generally pertains to data

performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring of said device's power consumption, comprising:	processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'. A clock attack is a procedure by which an attacker gains access to secure data or code used in a predetermined data processing routine being executed within a secure data processor, such as a secure microprocessor, by determining the time of execution of the predetermined data processing routine in relation to occurrence of an observable external event that precedes the predetermined routine . . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 1:37-51 – "The present invention prevents such clock attacks by providing a method that inhibits synchronization with externally generated instructions by preventing determination of the time of execution of a predetermined data processing routine in relation to occurrence of an observable external event that precedes execution of the predetermined routine. The method of the present invention includes the step of (a) randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine. 'Duration' is determined by the number of data processing clock cycles." 2:37-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the observable external event." 3:4-22 – "The preferred embodiments of the present invention are implemented in a secure microprocessor having a secure memory, a nonsecure memory and a secure central processing unit (CPU). Both memories may include both a random access memory (RAM) and a read only memory (ROM). The term –secure' indicates external inaccessibility. Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18; and further executes a maze of interim routines . . . assembled in response to instructions accessed from a secure RAM

	24." 3:27-31 – "In order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m." 8:1-22 – "The interim routine of Figure 5 includes the following subroutines, START, INITIALIZATION, iTH BIT OF X = 1?, SET iTH BIT OF Y TO 1, INCREMENT i, i > WORD SIZE OF SOURCE DATA?, DESTINATION DATA = Y ⊕ SOURCE DATA, WRITE NEW SECURE DATA, AND EXIT . . . .After the START subroutine 72, the CPU executes the INITIALIZATION subroutine 73, during which a register X is loaded with the result of the dynamic SOURCE DATA being XORed with SECURE DATA accessed from a secure RAM, a register Y is set to zero, and a counter i is set to zero." 9:31-34 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Claim 19 – "means for randomly varying the duration of said interim routines to thereby vary the duration between the occurrence of the externally observable event and the execution of the predetermined routine in order to prevent the time of execution of the predetermined data processing routine from being determined in relation to the occurrence of the externally observable event"
(a) receiving a variable amount of power, said power consumption varying measurably during said performance of said operation;	1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 2:39-47 – "The present invention further provides a data processing system, comprising means for executing a predetermined data processing routine, wherein a [sic] observable external event occurs in relation to execution of said predetermined routine; and means for randomly varying the duration between the occurrence of the observable external event and the execution of the predetermined routine to thereby prevent the time of execution of the predetermined data processing routine from being determined in relation to the

	occurrence of the observable external event." 9:18-23 – "The interim routine of Figure 5 need not be executed in combination with other interim routines and may be incorporated into an overall larger data processing routine anytime critical data needs to be moved from one location (source) to another location (destination)."
(b) receiving a quantity to be cryptographically processed, said quantity being representative of at least a portion of a message;	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines by a procedure known as a 'clock attack'." 3:11-17 – "Referring to Figure 1, in one preferred embodiment of the present invention, a CPU 10 executes a group of data processing routines . . . in response to instructions (code) accessed from a ROM 18 . . . ." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.
(c) introducing noise into said measurement of said power consumption while processing said quantity; and	1:54-2:2 – "In one aspect of the present invention, step (a) includes the steps of (b) executing one or more interim data processing routines between the occurrence of the observable external event and the execution of the predetermined routine; and (c) randomly varying the duration of said interim routines. In this aspect of the invention, steps (b) and (c) preferably include the step of (d) randomly assembling m said interim routines for said execution from a group of n stored routines having different durations, wherein m and n are integers, with n being greater than m. It is also preferred that step (d) either includes the step of (e) randomly accessing said m interim routines from a secure memory; or the steps of (f) randomly accessing pointers for said m interim routines from a secure memory; and (g) accessing said m interim routines from a memory in response to said pointers." 3:23-53 – "The routines ROUTINE N-1, ROUTINE N, ROUTINE N+1 are essential sequential subroutines of a repetitive overall larger data processing routine, and follow the occurrence of an observable external event during each sequence of the overall larger routine. In

	order to prevent a predetermined protected routine ROUTINE N from being synchronized with the observable external event that repetitively precedes the protected routine ROUTINE N, the BRANCH routine 12 causes the CPU 10 to branch to the maze of m interim routines INTERIM ROUTINE 1, INTERIM ROUTINE 2, ..., INTERIM ROUTINE m. The total duration of the m interim routines is a random variable." 3:64-4:6 – "The purpose of executing the interim routines is to provide a delay of a variable duration between ROUTINE N-1 and ROUTINE N. Each interim routine is a loop counter routine as shown in Figure 2, wherein the routine is completed once a number of clock cycles are counted. . . . After a START subroutine 25, the number of clock cycles D to be counted is loaded into a register implemented by the CPU 10 during a LOAD DELAY D IN R subroutine 27." 4:7-16 – "The number of clock cycles D is a random number accessed from the secure RAM 24. The random clock-cycle numbers are provided in the different portions of the secure RAM 24 from which the instructions for the different interim routines are accessed. The random clock-cycle numbers may be provided in the RAM 24 in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator." 6:53-58 – "The n interim routines have different durations; and the selection and sequence of the m interim routines that are assembled for execution during a given data processing sequence is randomly varied from one sequence to the next in order to make the total duration of the interim routines a random variable." 7:8-18 – "The pointers 67, 68, 69 are fixed but the selection and sequence of the m pointers that are accessed during a given data processing sequence is randomly varied from one overall data processing cycle to the next in order to make the total duration of the interim routines a random variable. The selection and sequence of the m pointers is provided during each overall data processing cycle in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."

(d) outputting said cryptographically processed quantity to a recipient thereof.	1:11-14 – "The present invention generally pertains to data processing and is particularly directed to preventing compromise of secure data processing routines. . . ." 1:29-34 – "An 'observable external event' is defined as any internal state transition that manifests itself externally, including but not limited to a change in voltage or current at any pin or combination of pins which is related to or affected by internal processor execution." 9:31-35 – "The method and system of the present invention are particularly suited for implementation in a microprocessor that controls descrambling of scrambled information signals by a descrambler in the possession of a subscriber in a subscriber communication network." Figures 1, 4. *See also* John F. Wakerly, MICROCOMPUTER ARCHITECTURE AND PROGRAMMING; THE 68000 FAMILY, John Wiley & Sons, Inc., New York, 1997 at 6-7.

Claim 30 ('661 Patent)	**U.S. 5,249,294 to Griffin**
The method of claim 29 wherein said step of introducing noise comprises: (a) generating initial noise having a random characteristic;	8:60-9:5 – "The total duration of the interim routine of FIG. 5 is dependent upon the number of ONE bits loaded into the X register during the INITIALIZATION subroutine 73, since the SET iTH BIT OF X TO 1 subroutine 75 is executed only when the ith bit of the X register equals ONE. Thus the range of variation in the duration of the interim routine of FIG. 5 is the number of clock cycles required for the subroutine 75 times the number of bit positions in the X register. Since the number of times that the subroutine 75 is executed is dependent upon the content of the dynamically processed source data and the secure data in the RAM, the total duration of the interim routine of FIG. 5 is a random variable that is unknown to the observer."
(b) improving the random characteristic of said initial noise; and	9:6-16 – "The randomness of the the total duration of the interim routine of FIG. 5 is further compounded by execution of the WRITE NEW SECURE DATA subroutine 79, which changes the secure data in the RAM that is accessed during the INITIALIZATION subroutine 73, so that each time the interim routine of FIG. 5 is executed, such secure data may be different. The secure data may be provided in the RAM in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number

	generator."
(c) varying said power consumption based on said improved initial noise.	9:6-16 – "The randomness of the the total duration of the interim routine of FIG. 5 is further compounded by execution of the WRITE NEW SECURE DATA subroutine 79, which changes the secure data in the RAM that is accessed during the INITIALIZATION subroutine 73, so that each time the interim routine of FIG. 5 is executed, such secure data may be different. The secure data may be provided in the RAM in response to a signal produced by a physically (truly) random phenomena, such as a noisy diode, or in response to a pseudorandom device, such as a pseudorandom number generator."